Práctica 6.4 Volúmenes en contenedores Docker

viernes, 21 de febrero de 2025 16:38

Creación de un volumen para una imagen de MySQL

1. Descarga y crea un volumen en Docker llamado *mysql_data.*

   Explicación:

   - docker volume create → Crea un volumen en Docker.
   - mysql_data → Es el nombre del volumen.

   Este volumen servirá para almacenar los datos de MySQL y asegurarnos de que persistan aunque eliminemos el contenedor.

```
ixchel@ixchel-Ubuntu:~$ sudo systemctl start docker
ixchel@ixchel-Ubuntu:~$ sudo systemctl enable docker
ixchel@ixchel-Ubuntu:~$ sudo docker volume create mysql_data
mysql_data
ixchel@ixchel-Ubuntu:~$
```

2. Ejecuta un contenedor de MySQL utilizando el volumen, configurando las credenciales y una base de datos de prueba.

   Explicación:

   - docker run -d → Ejecuta el contenedor en segundo plano.
   - --name mysql_container → Asigna el nombre mysql_container al contenedor.
   - -e MYSQL_ROOT_PASSWORD=rootpass → Define la contraseña del usuario root de MySQL.
   - -e MYSQL_DATABASE=testdb → Crea automáticamente una base de datos llamada testdb.
   - -e MYSQL_USER=testuser -e MYSQL_PASSWORD=testpass → Crea un usuario testuser con la contraseña testpass.
   - -v mysql_data:/var/lib/mysql → Monta el volumen mysql_data en la ruta donde MySQL guarda los datos (/var/lib/mysql).
   - mysql:latest → Usa la última versión de MySQL.

```
ixchel@ixchel-Ubuntu: ~
ixchel@ixchel-Ubuntu:~$ sudo docker run -d --name mysql_container \
> -e MYSQL_ROOT_PASSWORD=rootpass \
> -e MYSQL_DATABASE=testdb \
> -e MYSQL_USER=testuser \
> -e MYSQL_PASSWORD=testpass \
> -v mysql_data:/var/lib/mysql \
> mysql:latest
[sudo] contraseña para ixchel:
Unable to find image 'mysql:latest' locally
latest: Pulling from library/mysql
43759093d4f6: Pull complete
d255dceb9ed5: Pull complete
23d22e42ea50: Pull complete
431b106548a3: Pull complete
2be0d473cadf: Pull complete
f56a22f949f9: Pull complete
277ab5f6ddde: Pull complete
df1ba1ac457a: Pull complete
cc9646b08259: Pull complete
893b018337e2: Pull complete
Digest: sha256:146682692a3aa409eae7b7dc6a30f637c6cb49b6ca901c2cd160becc8112
Status: Downloaded newer image for mysql:latest
efa3971c1be9968c74b95a4ee10f04b914cff1436eda3c989ee550920e013208
ixchel@ixchel-Ubuntu:~$
```

3. Accede al contenedor y verifica que la base de datos ha sido creada.

   1. Entra al contenedor como root (-u root).
   2. Ingresa contraseña de root
   3. SHOW DATABASES;

   Explicación:

   - docker exec -it mysql_container → Ejecuta un comando dentro del contenedor mysql_container.
   - mysql -u root -p → Abre la consola de MySQL con el usuario root.
   - Luego, ingresamos la contraseña rootpass y verificamos las bases de datos con:
     - SHOW DATABASES;
   - Debería de aparecer testdb

```
ixchel@ixchel-Ubuntu: ~
ixchel@ixchel-Ubuntu:~$ sudo docker exec -it mysql_container mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 9.2.0 MySQL Community Server - GPL

Copyright (c) 2000, 2025, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
```

```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> SHOW DATABASES
    -> ;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| testdb             |
+--------------------+
5 rows in set (0.01 sec)

mysql>
```

4. Detén y elimina el contenedor de MySQL.

    1. Para salir del contenedor usa quit
    2. Deten y elimina el contenedor

    Explicación:

    - docker stop mysql_container → Detiene el contenedor.
    - docker rm mysql_container → Elimina el contenedor.

    Aunque eliminemos el contenedor, los datos aún estarán guardados en el volumen mysql_data.

```
ixchel@ixchel-Ubuntu:~$ sud
Enter password:
Welcome to the MySQL monito
Your MySQL connection id is
Server version: 9.2.0 MySQL

Copyright (c) 2000, 2025, O

Oracle is a registered trad
affiliates. Other names may
owners.

Type 'help;' or '\h' for he

mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| testdb             |
+--------------------+
5 rows in set (0.00 sec)

mysql> quit
Bye
ixchel@ixchel-Ubuntu:~$
```

```
ixchel@ixchel-Ubuntu:~$ sudo docker stop mysql_container
mysql_container
ixchel@ixchel-Ubuntu:~$ sudo docker rm mysql_container
mysql_container
ixchel@ixchel-Ubuntu:~$
```

5. Crea un nuevo contenedor reutilizando el volumen *mysql_data.*

    Explicación:

    - Se crea un nuevo contenedor llamado mysql_container2, pero **sin especificar la base de datos ni los usuarios**, porque **los datos ya están almacenados en el volumen** mysql_data.

```
ixchel@ixchel-Ubuntu:~$ sudo docker run -d --name mysql_container2 \
> -e MYSQL_ROOT_PASSWORD=rootpass \
> -v mysql_data:/var/lib/mysql \
> mysql:latest
[sudo] contraseña para ixchel:
acbffdeb9edc2df1517813ae8af13d5af745ca39c20ff9a0270e15044443c39b
ixchel@ixchel-Ubuntu:~$
```

6. Verifica que la base de datos sigue existiendo.

    Ejecutamos `SHOW DATABASES;` y confirmamos que `testdb` aún existe. Esto demuestra que los datos persistieron gracias al volumen.

```
ixchel@ixchel-Ubuntu:~$ sudo docker exec -it mysql_container2 mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 9.2.0 MySQL Community Server - GPL

Copyright (c) 2000, 2025, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
```

```
Owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| testdb             |
+--------------------+
5 rows in set (0.02 sec)

mysql>
```

7. Elimina el contenedor y el volumen *mysql_data.*

   Explicación:

   - docker stop y docker rm eliminan el nuevo contenedor.
   - docker volume rm mysql_data elimina el volumen, lo que borra los datos definitivamente.

```
ixchel@ixchel-Ubuntu:~$ sudo docker stop mysql_container2
mysql_container2
ixchel@ixchel-Ubuntu:~$ sudo docker rm mysql_container2
mysql_container2
ixchel@ixchel-Ubuntu:~$ sudo docker volume rm mysql_data
mysql_data
ixchel@ixchel-Ubuntu:~$
```

Creación de un volumen para una imagen de Nginx

1. Descarga y crea un volumen en Docker llamado *nginx_html.*

   nginx_html será el volumen que almacenará los archivos del sitio web.

```
ixchel@ixchel-Ubuntu:~$ sudo docker volume create nginx_html
nginx_html
ixchel@ixchel-Ubuntu:~$
```

2. Ejecuta un contenedor de Nginx utilizando el volumen, mapeando el puerto 8080 al 80 del contenedor.

   Explicación:

   - -p 8080:80 → Mapea el puerto 80 del contenedor al puerto 8080 del host. Así podemos acceder a Nginx en http://localhost:8080.
   - -v nginx_html:/usr/share/nginx/html → Monta el volumen nginx_html en la carpeta donde Nginx guarda las páginas web.

```
ixchel@ixchel-Ubuntu: $ sudo docker run -d --name nginx_container -p 8080:80 -v nginx_html:/usr/share/nginx/html nginx:l
atest
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
c29f5b76f736: Pull complete
e19db8451adb: Pull complete
24ff42a0d907: Pull complete
c558df217949: Pull complete
976e8f6b25dd: Pull complete
6c78b0ba1a32: Pull complete
84cade77a831: Pull complete
Digest: sha256:91734281c0ebfc6f1aea979cffeed5079cfe786228a71cc6f1f46a228cde6e34
Status: Downloaded newer image for nginx:latest
578d332fee05eb3f09a49f5b17ef4026d4deef066814a6166a26735a86561b08d
ixchel@ixchel-Ubuntu: $
```

3. Accede a http://localhost:8080 y verifica que la página de Nginx se muestra correctamente.

   Abrimos http://localhost:8080 y deberíamos ver la página de bienvenida de Nginx.

4. Modifica el contenido del sitio web dentro del volumen creando un nuevo archivo index.html.

   Creamos un nuevo archivo `index.html` en el volumen:

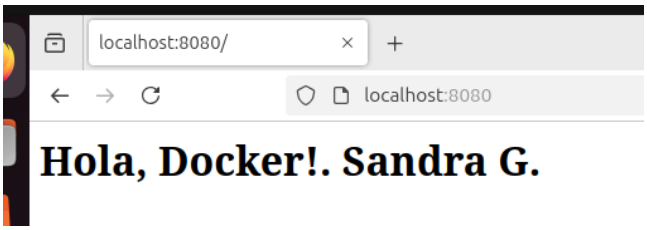
```
ixchel@ixchel-Ubuntu:~$ sudo docker run --rm -v nginx_html:/usr/share/nginx/html alpine sh -c 'echo "<h1> Hola, Docker!.
 Sandra G.</h1>" > /usr/share/nginx/html/index.html'
ixchel@ixchel-Ubuntu:~$
```

   Explicación:

   - Se usa alpine (una imagen ligera de Linux) para escribir un archivo index.html dentro del volumen.
   - El contenido del archivo será <h1>Hola, Docker!. Sandra G.</h1>.

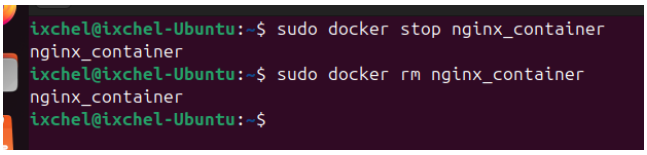
5. Recarga la página y verifica que los cambios se reflejan.

Visitamos http://localhost:8080 y vemos el mensaje, confirmando que el volumen almacena el contenido.



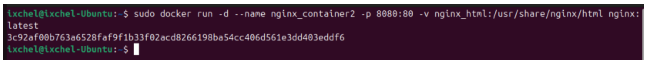
6. Detén y elimina el contenedor de Nginx.

   El contenedor se borra, pero los archivos siguen en el volumen.



7. Crea un nuevo contenedor reutilizando el volumen *nginx_html*.

   El nuevo contenedor sigue mostrando la página modificada, porque el archivo index.html está en el volumen nginx_html.

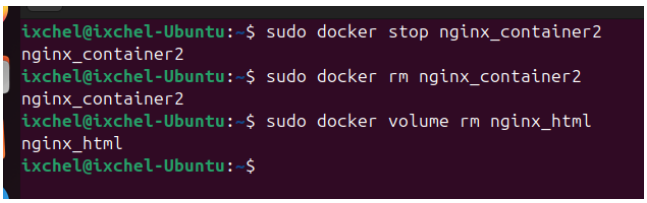


8. Verifica que los cambios en el contenido del sitio web persisten.

   Abrimos http://localhost:8080 y comprobamos

9. Elimina el contenedor y el volumen *nginx_html.*

   Esto borra el contenedor y elimina el volumen, lo que borra también el contenido web.



COMANDOS:

Creación de un volumen para MySQL

1. Descargar y crear un volumen llamado mysql_data

docker volume create mysql_data

2. Ejecutar un contenedor de MySQL con el volumen

docker run -d --name mysql_container -e MYSQL_ROOT_PASSWORD=root -e MYSQL_DATABASE=testdb -v mysql_data:/var/lib/mysql mysql:latest

3. Acceder al contenedor y verificar la base de datos

docker exec -it mysql_container mysql -uroot -proot -e "SHOW DATABASES;"

4. Detener y eliminar el contenedor

docker stop mysql_container

docker rm mysql_container

5. Crear un nuevo contenedor reutilizando el volumen

docker run -d --name mysql_container -e MYSQL_ROOT_PASSWORD=root -e MYSQL_DATABASE=testdb -v mysql_data:/var/lib/mysql mysql:latest

6. Verificar que la base de datos sigue existiendo

docker exec -it mysql_container mysql -uroot -proot -e "SHOW DATABASES;"

7. Eliminar el contenedor y el volumen

docker stop mysql_container

docker rm mysql_container

docker volume rm mysql_data

Creación de un volumen para Nginx

1. Descargar y crear un volumen llamado nginx_html

docker volume create nginx_html

2. Ejecutar un contenedor de Nginx con el volumen y mapear el puerto

docker run -d --name nginx_container -p 8080:80 -v nginx_html:/usr/share/nginx/html nginx:latest

3. Verificar que la página de Nginx se muestra correctamente

echo "Accede a http://localhost:8080 en tu navegador."

4. Modificar el contenido del sitio web

docker run --rm -v nginx_html:/usr/share/nginx/html busybox sh -c 'echo "<h1>Hola desde Docker Volumes</h1>" > /usr/share/nginx/html/index.html'

5. Recargar la página y verificar los cambios

echo "Recarga http://localhost:8080 en tu navegador para ver los cambios."

6. Detener y eliminar el contenedor

docker stop nginx_container

docker rm nginx_container

7. Crear un nuevo contenedor reutilizando el volumen

docker run -d --name nginx_container -p 8080:80 -v nginx_html:/usr/share/nginx/html nginx:latest

8. Verificar que los cambios persisten

echo "Recarga http://localhost:8080 en tu navegador para confirmar que los cambios se mantienen."

9. Eliminar el contenedor y el volumen

docker stop nginx_container

docker rm nginx_container

docker volume rm nginx_html